REPORT DOCUMENTATION PAGE

Form Approved
OMB NO. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE 01 DEC 00	3. REPORT TYPE AND DATES COVERED Final Report: JAN 00 - DEC 00

4. TITLE AND SUBTITLE
ACE LINK - An Approach to Integrating Command and Control Model Architecture

5. FUNDING NUMBERS
C
F33615-00-C-1669

6. AUTHOR(S)
Woodring, John W.; Kashmiri, Rafiah
Navarro, Guillermo

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Modasco, Inc.
4303 Vineland Road, Suite F-7
Orlando, FL 32811

8. PERFORMING ORGANIZATION REPORT NUMBER
FR1

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U. S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
USAF/AFMC
AIR FORCE RESEARCH LAB
2310 EIGHT STREET, BUILDING 167
WRIGHT-PATTERSON AFB, OHIO
45433-7801

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

12 a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

12 b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Report developed under SBIR contract for topic AF00-116.
A software package that provides powerful tools to assist engineers in designing and analyzing complex systems is proposed. This package allows multiple designers, working independently in a collaborative environment, to create a digital design of a process in individual modules, bind them together and simulate their performance under internal and external control. The resulting virtual design is tied to a database of system requirements and automatically responds to requirement changes. System designs are developed as Colored Petri Nets using a graphical editor supported with tools to create, modify, edit, store and merge designs. Other tools help the designer describe the system's behavior and relate it to both objects defined and stored in a database and to external modules from legacy or ancestral simulations. Analysis and evaluation tools are provided to execute large numbers of test scenarios, measure system-performance metrics and identify and report failures and malfunctions. This methodology is robust with respect to describing concurrent events arising from multiple external controls and accurately models the behavior of distributed-processing systems. The design tools have the power and generality to model and simulate virtually all commercial and military processes, including those related to human performance in a combat environment.

14. SUBJECT TERMS
SBIR Report
Modeling and Simulation; Business Engineering;
Process Modeling; Command and Control Architecture;
Collaborative Virtual Prototyping

15. NUMBER OF PAGES
131

16. PRICE CODE

17. SECURITY CLASSIFICATION OR REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION ON THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500

Standard Form 298 (Rev.2-89)
Prescribed by ANSI Std. 239-18
298-102

20010301 079

ACE Link - An Approach to Integrating
Command and Control Model Architectures
Final Report
January 15, 2001

Prepared by

Modasco Inc.
4303 Vineland Road, Suite F-7
Orlando, Florida 32811
and
Emergent Information Technology - East
1700 Diagonal Road Suite 500
Alexandria, VA 22314

For

Department of the Air Force
Air Force Research Laboratory
Wright-Patterson Air Force Base, Ohio 45433

Under Contract
F33615-00-C-1669

Table of Contents

Executive Summary

The Modasco-EITE team investigated the current capability of LEdit, a graphical-design tool developed by the U.S. Air Force Electronic System's Center (ESC), to assist analysts in describing complex relationships. LEdit uses Colored Petri Nets (CPNs) as its design methodology and provides a mature set of features that can be applied to a wide variety of problems that occur in military and commercial applications. Two distinct commercial scenarios have been considered: (1) LEdit can be used as either a stand-alone tool, or can be integrated into other design and reporting tools, to graphically describe the complex relationships between analysis elements. (2) LEdit can be integrated into predictive models as a graphical design tool. In this case, LEdit would either be modified, or middleware developed, to provide input to the analysis engine of the model.

This report is divided into four sections:
1. A description of the opportunity and approach
2. An interface demonstration
3. The application to THUNDER
4. A THUNDER demonstration

1. Introduction

Modasco was provided an executable version of LEdit by the Government soon after the project start date of 8 May 2000. The ACE Link development team members from Modasco, Inc. included Rafiah Kashmiri as Program Manager and Dr. John Woodring as the principle investigator of the ACE Link architecture. Mr. Greg Jablunovsky of Emergent Technology was responsible for the THUNDER integration.

2. A Description of the Opportunity and Approach

The Air Force has developed powerful tools for designing complex systems, but there is no direct way of evaluating the military worth of a proposed design. The tool presently used to design the architecture is LEdit. THUNDER is the tool used to evaluate combat models. ACE-Link will attempt to link the architecture with the combat evaluation to provide a direct way of evaluating the military worth of a proposed design.

2.1 LEdit

LEdit is a reasonably mature software tool that assists the user in describing generic processes. LEdit is part of the MRT toolset used to graphically describe Command and Control (C2) architecture. It uses Colored Petri Nets as its' design language. However, while simple to use and powerful in its ability to clarify a complex system, its use has been, up until this time, limited to **qualitative** analysis.

One of LEdits most ubiquitous uses is Time Critical Targeting (TCT). A TCT network designed using LEdit may describe a variety of items. The resulting graphical design can illustrate the various communication paths between military components. Each path in the network can be distinguished using a particular pre-determined color. This network graph will impart upon a viewer two things: 1) That there are many possible communication links between military components; and 2) For any single communication component to be operable, all of its intermediate communication nodes must also be operable.

However, there are current limitations to the usefulness of LEdit. The network graph will fail when you attempt to derive information regarding optimal paths based upon minimal communications time delay if, for example, all the communication nodes are operable or even if one or more of the nodes is inoperable. In this way, LEdit and TCT fail the user.

2.2 THUNDER

THUNDER is a simulation of theater warfare used by modelers to quantitatively determine how and why various mission outcomes occur. It models Time Critical Targeting in an unrealistic and oversimplified way.
There is no specification of individual communications nodes or communications links. Therefore, the model is not sensitive to realistic events such as:
- A node that has not yet been established in the eventTime delays that are specific to the individual nodesA node that has been partially or totally destroyed

The critical question yet remains: How to integrate the graphical design capability of LEdit with the simulation capability of THUNDER?

3.0 ACE LINK – A Prototype Solution

ACE Link is a prototype solution that seeks to integrate the network graphing ability of LEdit with THUNDER's evaluative processes. ACE Link is middleware that sits between LEdit and THUNDER and creates a synthesis of the two tools. It accomplishes this using a high level linking structure. ACE Link receives data, optimizes the communications paths and returns it to a buffer. THUNDER++ retrieves the data from the buffer, administers an evaluative process that then specifies the appropriate network.

We implement ACE Link through the following four tasks:
1. Embedding time delays in the LEdit design
2. Constructing an Input File
3. Executing ACE Link
4. Reading the file created by ACE Link

3.1 Embedding time delays in the LEdit design

Time delays are embedded into the LEdit design by first opening the LEdit graphs. Individual path segments are selected by double clicking on them. Time delays are entered for each communications-path segment in the "Edit edge" Dialog Box. Finally, these changes are saved in the system by clicking on the "OK" button.

3.2 Constructing an Input File

An input file is created by THUNDER. THUNDER performs this action, generating a file that contains the LEdit file name, the transmitter node, the receiver node and a listing of all non-operational nodes.
The resulting code resembles this:
> LEdit File Name:
> *(some transmitting node, for example \tct_2.edt)*
> Transmitter Node:
> *(some transmitting node, for example a U2)*
> Receiver Node:
> *(some transmitting node, for example a Airborne Weapons System)*
> Non-Operational Nodes:
> *(some non-operational nodes)*

3.3 Executing ACE Link

The following steps are undertaken when executing ACE Links. Double click on the ACE Link icon. ACE Link will then:
- Open the LEdit File Specified in Line 2 of the Input File
- Read the Name of the "Transmitter" Node Specified in Line 4 of the Input File
- Read the Name of the "Receiver" Node Specified in Line 6 of the Input File
- Read the Names of the "Inoperable" Nodes on Line 8.. (None in this case)
- Find the Communications Paths With the Minimum Time Delay Subject to These Conditions
- Create the Output File for THUNDER to Read
- (Enter 0 as required to terminate ACE Link Execution)

3.4 Reading the file created by ACE Link

ACE Link will then generate a text message that will identify the optimum path, the node sequence for this particular path, and the delay time. The following is sample output:

> Minimum Time Delay = 33
> U2
> CARS
> Decision Maker
> RITA
> Airborne Weapons Systems

This is the absolute optimized path. If for example non-operable nodes were designated in the initial input file, the optimized communication path would be different. For example, if node RITA was declared non-operable in the input file below:
LEdit File Name:

(some transmitting node, for example \tct_2.edt)
Transmitter Node:
(some transmitting node, for example a U2)
Receiver Node:
(some transmitting node, for example a Airborne Weapons System)
Non-Operational Nodes:
(some non-operational node, for example RITA)

then the resulting optimized output would be:

> Minimum Time Delay = 36
> U2
> CARS
> Decision Maker
> AWACS
> Airborne Weapons Systems

4. The Significance of ACE Link

4.1 Ability to Quantify a System Design

Prior to ACE Link, LEdit was a One-Dimensional Tool. It provided only a qualitative picture of the system. Data describing the system was not encapsulated in the design. Analysis of the design had to be performed "Elsewhere" - in other combat simulation models such as THUNDER. This is problematic for the following reason:
It is virtually impossible to re-use an LEdit design in a new model
For ACE Link to be generally accepted, it must be related to object oriented analysis & design. The modern view of simulation and modeling is based upon the ideas of OOAD. The Model Should:

- Be Designed Visually
  Example: CPN Model Created by LEdit

- Encapsulate Data Describing the Design
  Example: Communications Time Delays

- Provide Interactive Services to Answer Questions About the System
  Example: What is the Optimum Path Subject To a Set of Constraints.

There are several Advantages to this approach.Once a system has been designed, it can be implemented In a number of different simulation models (THUNDER, STORM..)

- Modification of the system is simplified, since the attributes and methods of the system are encapsulated In It.

- This methodology directly supports object oriented software development

- This methodology also directly supports RDB interfaces

7

5. Conclusions

5.1   Summary of Work Accomplished

Much effort has been expended in creating models that have sufficient realism to infer the probable outcome of a real battle. As each new generation of models is developed, more detail is added to each aspect of the battle with the expectation of more reliable results. In most cases, the increased realism results directly from better data, often obtained through testing and sometimes obtained from the outcomes of military actions. However, the cost of obtaining even an incremental improvement in realism is increased by the concurrent changes in the model's architecture made to improve execution time, software maintainability and the analyst's interface with the model. While these are all worthy features, they must be obtained by an almost complete re-write of the model, and, from the view-point of those who procure such developments, it must seem as if most of the cost and development time is expended in re-doing work already bought and paid for. What is clearly needed is a standard set of re-usable models of battlefield subsystems that can be glued together to simulate modern warfare. The work performed by Modasco in Phase I has demonstrated the feasibility of creating such stand-alone modules, which we refer to as **Simulation Objects**. Integrating this technology into mission simulation tools such as EADSIM will greatly improve the way analysts model complex communications systems. As a part of the JCAPS toolkit, it will have an immediate impact on defining how the next generation of military-worth models will be developed. Currently, the lack of commonality in the way the Army, Navy and Air Force model the evolution of a battle hinders attempts to develop a comprehensive model of a modern war; providing a standard set of development tools that includes Visual Simulation Objects will provide this commonality. Simulation Objects are very closely related to the class instantiations of Object Oriented Analysis and Design (OOAD). However, they extend the ideas of OOAD in several important ways:

- They add a visual aspect to the design. Designing systems graphically is generally faster and leads to fewer errors.

- They act as a repository of information and can respond to queries from other Simulation Objects.

- They behave dynamically, simulating the behavior of the real objects they represent.

In Phase I, Modasco developed the first Simulation Object, which we named ACE Link (Architecture - Combat Evaluation Link). The Architecture component of ACE Link was LEdit, a legacy tool developed at the Electronic Systems Center (ESC). LEdit provides a graphical-design interface based upon the use of Colored Petri Nets (CPNs) and has been used extensively to design and evaluate Command and

8

Control (C2) systems. THUNDER provides the dynamic model behavior of ACE Link. As a component of the Air Force analytic toolkit, it measures force-level operational impact, primarily through Monte Carlo iterations, including Time Critical Targeting (TCT). Modasco developed a third software module to complete ACE Link. Consisting of approximately 1,000 lines of C++ source code, this module provides an interface between LEdit and THUNDER and responds to queries from THUNDER to identify the optimal communications path between a missile-site detector and attacker.

The ACE Link synthesis of LEdit, THUNDER and the data repository-query capability introduced by Modasco provides a significant improvement to the realism of models that include TCT.

Before ACE Link, THUNDER modeled the time delay between missile-site detection and attack using a single, fixed time delay for all communications paths. The same time delay was used independent of whether one or more communications nodes were inoperable, either because the node had not been installed or it had been damaged during the battle. This provided an over-simplified picture of the battle's evolution from missile-site detection to attack. One would expect larger time delays early in the battle before all of the communications nodes had been installed and during the last stages of an unsuccessful battle when many nodes were damaged by enemy action. The result of using an average time delay is the elimination of the result (i.e., impaired communications) of an enemy action and thus an over-estimation of your ability to continue a battle under duress. In actual warfare, the effect of loosing, or significantly delaying, communications leads to a critical state in which the battle can no longer be conducted in an "average" way, and attacks on missile sites will be eliminated rather than reduced. The error is in modeling a clearly non-linear event (TCT) as a linear one.

ACE Link provides a more realistic picture of TCT by explicitly taking into account the actual status (operable or inoperable) of all of the communications nodes. This capability was added without modifying the source code of LEdit and with only very minimal changes to THUNDER. The improved capability was implemented in the following way:

- The systems analyst embeds within the LEdit graph the time delay for each communications path between nodes. This is done using existing LEdit capability. When LEdit generates the output file for this communications system, these time delays are included within the textual information contained in the file.

- THUNDER uses a Neural Network (NN) to provide rapid estimations of the TCT time delay during a simulation event. The "training" of the NN occurs before the initiation of the event. THUNDER creates a table of communications paths between pairs of nodes and generates "cases" in which one or more of the nodes is inoperable. For each case, it

queries ACE Link to find the optimal communications path consistent with the constraint that the specified nodes are inoperable. It then stores the result in the NN database for use during a simulation event.

- The ACE Link module binds the LEdit-designed TCT model to THUNDER by acting as middleware, responding to queries from THUNDER and extracting data from LEdit. When THUNDER requests the time delay for the optimal communications path, ACE Link calculates its value and responds to the request. This calculation is greatly simplified by LEdit's capability of allowing the system designer to enumerate all possible communications paths and to mark them with color (the "color" in Colored Petri Nets). Thus, ACE Link must only parse the colored "loops" identified by the designer. This makes the identification of the optimal communications path very rapid and thus applicable to a variety of real-time applications.

5.1.1 Application to THUNDER

The Modasco-EITE team applied ACE Link to THUNDER to demonstrate the interface between an architecture design tool and a combat evaluation model. The following steps were performed:

a. The number of disabled nodes was randomly chosen from a Gaussian distribution with a user specified mean and variance. Let the number of disabled nodes be N_d and the total number of nodes be N.

b. For each node, other than the starting and terminating node, determine whether it is disablerd by the following prescription:

   i. Calculate the fraction of nodes disabled as N_d/N.
   ii. Select a random real number, R, from a uniforn random distribution on [0,1] for each node.
   iii. If $R < N_d/N$, disable the node.

c. Calculate the path with the smallest latency

Figure 1 shows the operational scenario designed using LEdit. Figure 2 shows results from the integration of ACE Link and THUNDER.

5.2   Significance of ACE Link

While the architecture of ACE Link is not optimal, its limitations were imposed by two constraints: (1) The need to rapidly prototype a Simulation Object demonstrating the

power and utility of the concept and (2) The requirement to not modify the LEdit source code in the Phase I investigation. ACE Link in its broadest sense encompasses LEdit, THUNDER and the ACE Link module itself. Its significance is based upon its ability to link a design architecture (LEdit) and a simulation architecture (THUNDER) and to add a data repository and data queuing capability. More importantly, it demonstrates that new capability can be bound to legacy programs with minimal intrusion by using the Simulation Objects concept. The generalization of Simulation Objects is discussed in paragraph 1.4 of this proposal, where it is shown to be applicable to both military and commercial activities.
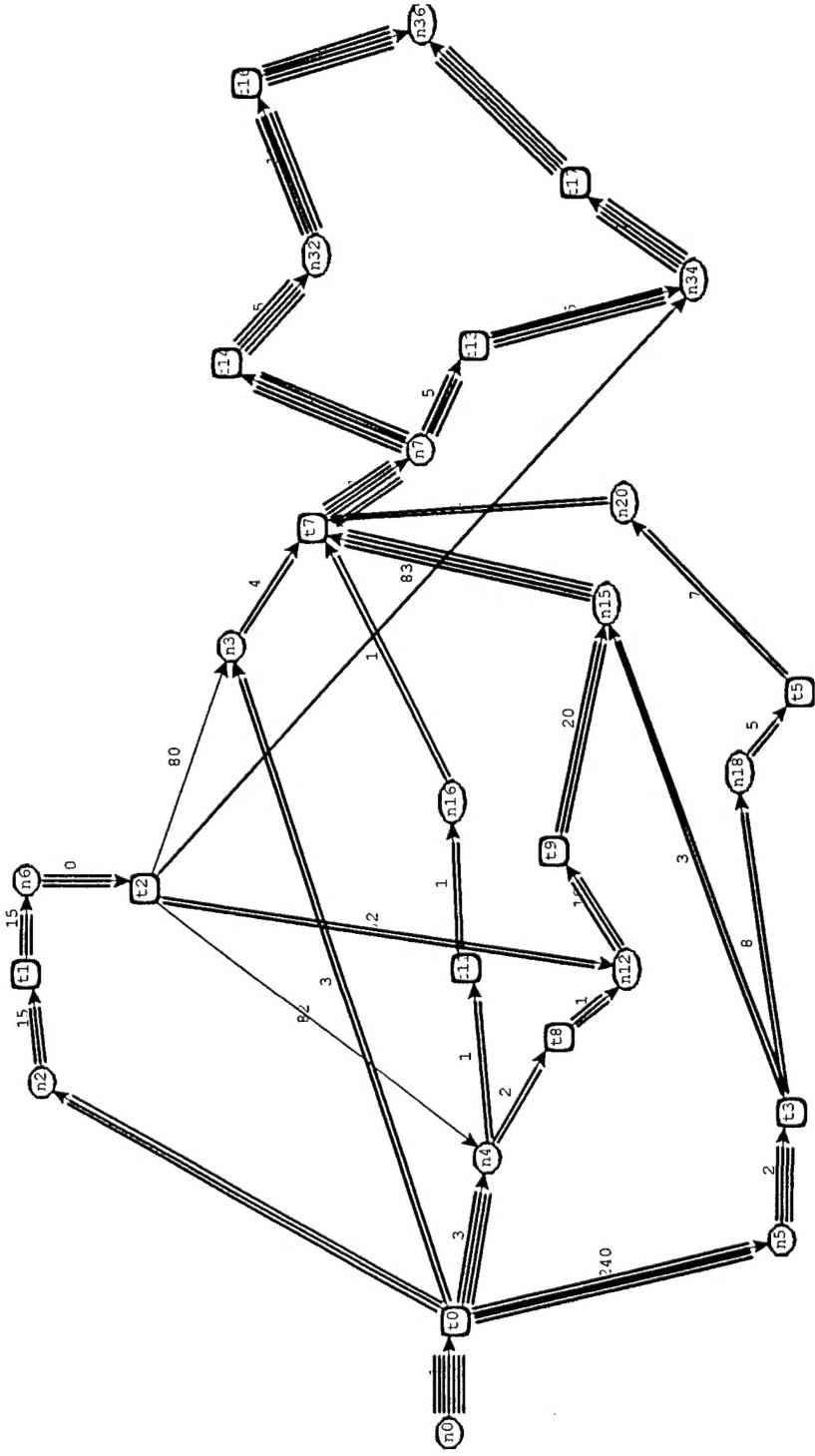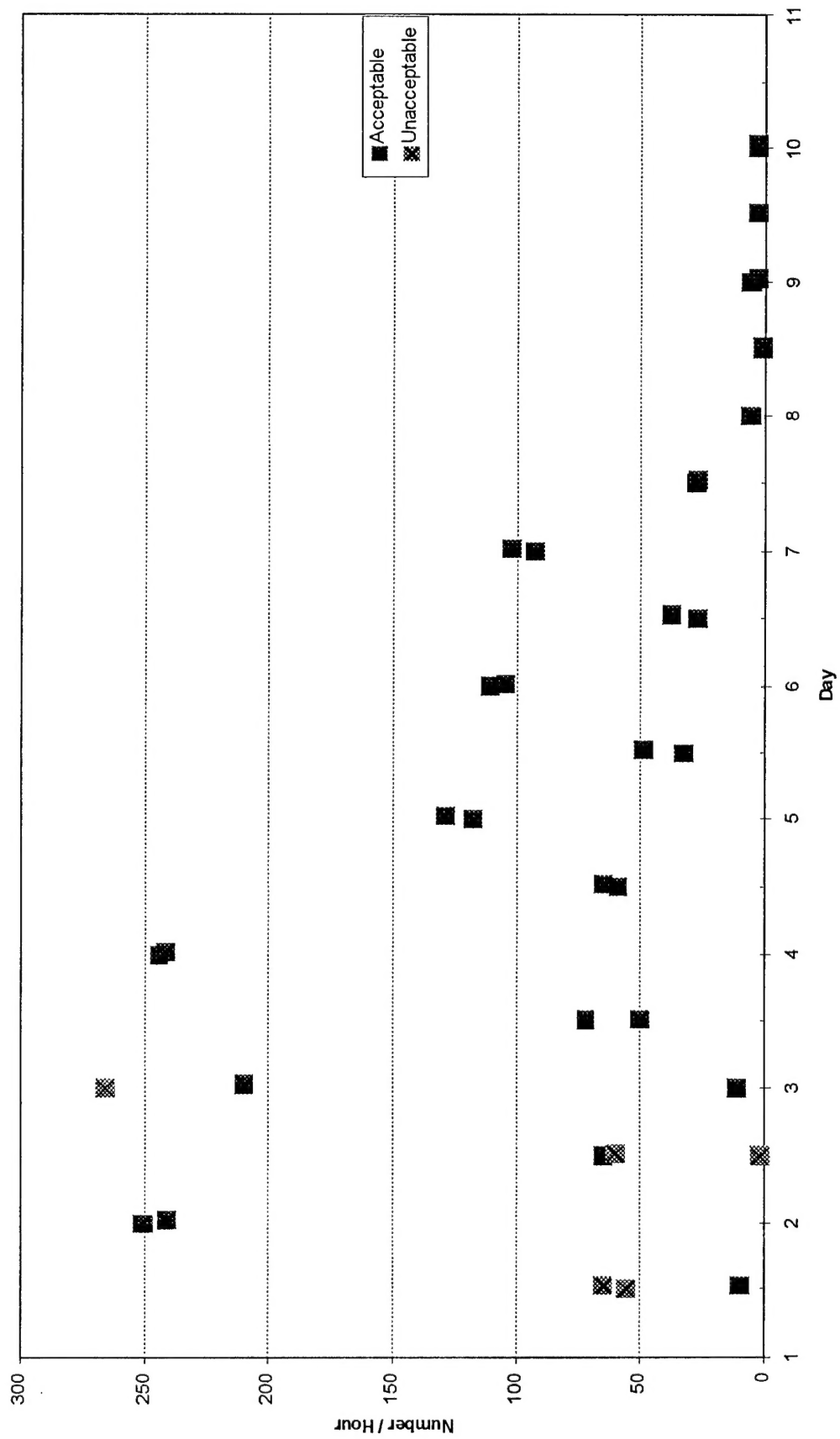
Figure 1. Time Critical Targeting

Figure 2. Latency Distribution. The vertical axis represents the number of Launch Detection Events per Hour and the horizontal axis represents the scenario time in days. The threshold latency is 5 minutes.

13

5.3　Lessons Learned

Three features of a Simulation Object have been identified. ACE Link demonstrates that designing systems using LEdit meets the goal of model visualization very effectively. Additionally, its loop-designation feature greatly enhances the dynamic behavior of the object by providing a rapid means of parsing all the possible model paths. The Phase II development should thus incorporate LEdit into the Simulation Objects architecture.

THUNDER, a legacy simulation model, was used only as a demonstration of how ACE Link binding could be achieved. The power of ACE Link lies in it ability to bind subsystem models together to form a larger, more comprehensive simulation. Thus, binding should be a generalized feature of the development that can be used to attach a new capability to an existing capability. While THUNDER served its purpose in Phase I, it is not a good candidate for demonstrating Simulation Object binding in Phase II.

Querying ACE Link for TCT optimal communications path data was achieved by a methodology dictated by the THUNDER design. THUNDER's ability to communicate with other programs is limited. Thus, THUNDER writes its request (including communications node outages) to a text file, and ACE Link responds by writing the result to a second text file, which subsequently is read by THUNDER. This somewhat cumbersome communications systems was devised ad hoc to meet the constraints of the THUNDER architecture, but is not powerful enough for use in Phase II. Communications between two executable programs may be required for legacy systems that cannot fully link to Simulation Objects. This will be referred to as weak binding, to contrast to the strong binding inherent in Simulation Objects. Binding a new Simulation Object to an assemblage of other Simulation Objects is discussed above.